Amendments to the Claims:

This listing of claims will replace all prior versions and listings of claims in this application:

Listing of Claims:

1. (Currently amended) A processor-implemented method of efficiently writing records from an in-memory database to a disk database, comprising:

linking the records by a linked list;

creating a header data structure of the linked records;

linking a new record in the in-memory database to the header data structure; ~~and~~

transferring the records in the linked list and the new record from the in-memory database to the disk database using the header data structure;

wherein transferring the records includes transferring blocks of records as a single transaction to the disk database, in order to minimize the number of transfers from the in-memory database to the disk database;

dynamically setting a last commit pointer and a last flush pointer to keep track of: (i) a block of records that have been transferred to the disk database but not yet committed; (ii) a block of records that have not been transferred to the disk database; and (iii) a block of records that have been transferred and committed to the disk database, in order to maintain synchronization between the in-memory database and the disk database; and

determining whether a committing operation of the block of records

that have been transferred to the disk database but not yet committed is successful, and if the committing operation is determined to be unsuccessful, deeming the committing operation of all the records in the block of records that have been transferred to the disk database but not yet committed is successful, to have failed.

2. (Original) The method of claim 1, wherein each record contains information about a single transaction.

3. (Original) The method of claim 1, wherein the header data structure comprises an entity name that identifies the record.

4. (Original) The method of claim 3, wherein the header data structure further comprises a last commit pointer.

5. (Original) The method of claim 4, wherein the header data structure further comprises a last flush pointer.

6. (Original) The method of claim 5, wherein the header data structure further comprises a head pointer.

7. (Original) The method of claim 6, wherein the header data structure further comprises a tail pointer.

8. (Original) The method of claim 7, further comprising locating a new header data structure for the new record

9. (Original) The method of claim 8, further comprising determining if the head pointer in the header data structure of the new record points to another record.

10. (Original) The method of claim 9, wherein if the head pointer in the header data structure of the new record points to another record getting the other record.

11. (Canceled)

12. (Original) The method of claim 7, further comprising determining if the last commit pointer of the header data structure points at a record.

13. (Original) The method of claim 12, further comprising setting the last commit pointer equal to the head pointer if the last commit pointer does not point to a record.

14. (Original) The method of claim 12, further comprising getting the record to which the last commit pointer points.

15. (Original) The method of claim 14, further comprising determining whether there is a last record after the record to which the last commit pointer points.

16. (Original) The method of claim 15, further comprising writing to

the disk memory all records in the link list that occur after the record to which the last commit pointer points.

17. (Original) The method of claim 16, further comprising setting a last flush pointer of the header data structure equal to the last count if the writing of the records to the disk memory ended successfully.

18. (Original) The method of claim 17, further comprising setting the last flush pointer equal to the last commit pointer if the writing of the records to the disk memory did not end successfully.

19. (Currently amended) A computer program product having instruction codes stored on a computer-usable medium for efficiently writing records from an in-memory database to a disk database, comprising:

a ~~first~~ set of instruction codes for linking the records by a linked list;

a ~~second~~ set of instruction codes for creating a header data structure of the linked records;

a ~~third~~ set of instruction codes for linking a new record in the in-memory database to the header data structure; and

a ~~fourth~~ set of instruction codes for transferring the records in the linked list and the new record from the in-memory database to the disk database using the header data structure;

wherein the set of instruction codes for transferring the records includes a set of instruction codes for transferring blocks of records as a single transaction to the disk database, in order to minimize the number of transfers from the in-

memory database to the disk database;

a set of instruction codes for dynamically setting a last commit pointer and a last flush pointer to keep track of: (i) a block of records that have been transferred to the disk database but not yet committed; (ii) a block of records that have not been transferred to the disk database; and (iii) a block of records that have been transferred and committed to the disk database, in order to maintain synchronization between the in-memory database and the disk database; and

a set of instruction codes for determining whether a committing operation of the block of records that have been transferred to the disk database but not yet committed is successful, and if the committing operation is determined to be unsuccessful, deeming the committing operation of all the records in the block of records that have been transferred to the disk database but not yet committed is successful, to have failed.

20. (Original) The computer program product of claim 19, wherein each record contains information about a single transaction.

21. (Original) The computer program product of claim 19, wherein the header data structure comprises an entity name that identifies the record.

22. (Original) The computer program product of claim 21, wherein the header data structure further comprises a last commit pointer.

23. (Original) The computer program product of claim 22, wherein the

header data structure further comprises a last flush pointer.

24. (Original) The computer program product of claim 23, wherein the header data structure further comprises a head pointer.

25. (Original) The computer program product of claim 24, wherein the header data structure further comprises a tail pointer.

26. (Currently amended) A processor-implemented system for efficiently writing records from an in-memory database to a disk database, comprising:

~~a first set of instruction codes~~ means for linking the records by a linked list;

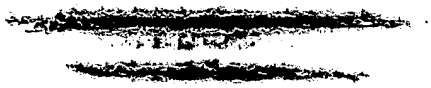
~~a second set of instruction codes~~ means for creating a header data structure of the linked records;

~~a third set of instruction codes~~ means for linking a new record in the in-memory database to the header data structure; ~~and~~

~~a fourth set of instruction codes~~ means for transferring the records in the linked list and the new record from the in-memory database to the disk database using the header data structure;

wherein the means for transferring the records includes a means for transferring blocks of records as a single transaction to the disk database, in order to minimize the number of transfers from the in-memory database to the disk database;

means for dynamically setting a last commit pointer and a last flush pointer to keep track of: (i) a block of records that have been transferred



to the disk database but not yet committed; (ii) a block of records that have not been transferred to the disk database; and (iii) a block of records that have been transferred and committed to the disk database, in order to maintain synchronization between the in-memory database and the disk database; and

means for determining whether a committing operation of the block of records that have been transferred to the disk database but not yet committed is successful, and if the committing operation is determined to be unsuccessful, deeming the committing operation of all the records in the block of records that have been transferred to the disk database but not yet committed is successful, to have failed.

27. (Original) The system of claim 26, wherein each record contains information about a single transaction.

28. (Original) The system of claim 26, wherein the header data structure comprises an entity name that identifies the record.

29. (Original) The system of claim 28, wherein the header data structure further comprises a last commit pointer.

30. (Original) The system of claim 29, wherein the header data structure further comprises a last flush pointer and a head pointer.